What Is Claimed Is:

1. A method of managing a cache comprising:

identifying program states associated with an executing program;

comparing a time of first discovery of a program state to a most recent time of first discovery to determine if the program state is associated with short-lived objects; and

if the program state is not associated with short-lived objects and program execution has transitioned to the program state from a program state associated with short-lived objects, flushing the cache of at least one short-lived object.

2. A method as defined in claim 1 wherein identifying the program states associated with the executing program comprises:

developing a trace of the program; and

identifying the program states from the trace by comparing sets of data at least partially indicative of addresses associated with the trace.

3. A method as defined in claim 2 wherein the sets of data are intersecting sets of data.

4. A method as defined in claim 2 wherein the addresses are memory addresses.

5. A method as defined in claim 1 wherein the most recent time of first discovery is associated with a most recently discovered program state.

6. A method as defined in claim 1 wherein comparing the time of first discovery of the program state to the most recent time of first discovery comprises comparing the time of first discovery of the program state to a percentage of the most recent time of first discovery.

7. A method as defined in claim 1 further comprising comparing a time of first discovery of a previous program state proceeding the program state to the most recent time of first discovery to determine if program execution has transitioned from a program state associated with short-lived objects to a program state associated with long-lived objects.

8. A method as defined in claim 7 wherein there is no program state between the program state and the previous program state.

9. A method as defined in claim 7 wherein comparing the time of first discovery of the previous program state to the most recent time of first discovery comprises comparing the time of first discovery of the previous program state to a percentage of the most recent time of first discovery.

10. A method as defined in claim 1 wherein flushing the cache comprises releasing at least a portion of the cache associated with the at least

one short-lived object for overwriting.

11. A method as defined in claim 1 wherein flushing the cache comprises pre-fetching data associated with program states having a time of first discovery less than a predetermined percentage of the most recent time of first discovery.

12. A method as defined in claim 1 wherein flushing the cache comprises pre-fetching data associated with program states (a) having a time of first discovery less than a predetermined percentage of the most recent time of first discovery and (b) meeting a usage criterion.

13. An article of manufacture storing machine readable instructions which, when executed, cause a machine to:

identify program states associated with an executing program;

compare a time of first discovery of a program state to a most recent time of first discovery to determine if the program state is associated with short-lived objects in a cache; and

to flush the cache of at least one short-lived object if the program state is not associated with short-lived objects and program execution has transitioned to the program state from a program state associated with short-lived objects.

14. An article of manufacture as defined in claim 13 wherein the most recent time of first discovery is associated with a most recently discovered program state.

15. An article of manufacture as defined in claim 13 wherein the instructions cause the machine to compare the time of first discovery of the program state to the most recent time of first discovery by comparing the time of first discovery of the program state to a percentage of the most recent time of first discovery.

16. An article of manufacture as defined in claim 15 wherein the instructions further cause the machine to compare a time of first discovery of a previous program state proceeding the program state to the most recent time of first discovery to determine if program execution has transitioned from a program state associated with short-lived objects to a program state associated with long-lived objects.

17. An article of manufacture as defined in claim 15 wherein the instructions further cause the machine to compare a time of first discovery of a previous program state proceeding the program state to a percentage of the most recent time of first discovery to determine if program execution has transitioned from a program state associated with short-lived objects to a program state associated with long-lived objects.

18. An apparatus to manage a cache comprising:

a trace sampler to develop a trace of a program;

a state identifier to identify program states from the trace; and

a short-lived object identifier to identify program states associated with short-lived objects based on the times of first discovery of the program states.

19. An apparatus as defined in claim 18 further comprising a cache flusher to remove a short-lived object from the cache when the executing program transitions from a program state associated with the short-lived object to a program state associated with a long-lived object.

20. An apparatus as defined in claim 19 wherein the cache flusher removes the short-lived object by releasing a memory location where the short-lived object is stored for overwriting.

21. An apparatus as defined in claim 19 wherein the cache flusher removes the short-lived object by pre-fetching an object associated with a program state having a first discovery time less than a predetermined percentage of the most recent discovery time.

22. An apparatus as defined in claim 21 further comprising a usage filter to limit the objects pre-fetched by the cache flusher to program states meeting a usage criterion.

23. An apparatus as defined in claim 18 wherein the trace comprises a memory address trace.

24. An apparatus as defined in claim 18 wherein the state identifier further comprises:

a signature developer to develop possible state signatures for sets of entries in the trace; and

a state distinguisher to identify program states based on the possible state signatures.

25. An apparatus as defined in claim 24 wherein the signature developer develops a first possible state signature by mapping a first set of entries in the trace to a first bit vector signature.

26. An apparatus as defined in claim 25 wherein the signature developer develops a second possible state signature from a second set of entries in the trace by mapping the second set of entries to a second bit vector signature.

27. An apparatus as defined in claim 26 wherein the first set of entries partially intersects the second set of entries.

28. An apparatus as defined in claim 18 wherein at least one of the state identifier and an entropy calculator indexes the program states by their

respective times of first discovery.

29. An apparatus as defined in claim 18 wherein the short-lived object identifier identifies a program state as being associated with a short-lived object by comparing a time of first discovery of the program state to a most recent time of first discovery.

30. An apparatus as defined in claim 29 wherein a cache flusher removes the short lived object from the cache if the program state is associated with the short-lived object and a time of first discovery of a following program state is less than a threshold.

31. An apparatus as defined in claim 18 wherein the short-lived object identifier identifies the program state as being associated with a short-lived object if the time of first discovery of the program state is greater than a predetermined percentage of the most recent time of first discovery.

32. An apparatus as defined in claim 31 wherein a cache flusher removes the short lived object from the cache if the program state is associated with the short-lived object and the time of first discovery of a following program state is less than the predetermined percentage of the most recent time of first discovery.

33. A method of managing a cache comprising:

identifying program states associated with an executing program; and

if an age of a current program state is less than a first threshold and an age of a previous program state is greater than a second threshold, pre-fetching memory objects associated with program states having an age less than a third threshold.

34. A method as defined in claim 33 wherein identifying the program states associated with the executing program comprises:

developing a trace of the program; and

identifying the program states from the trace by comparing sets of data at least partially indicative of addresses associated with the trace.

35. A method as defined in claim 33 wherein the first threshold comprises a percentage of first discovery time of a most recently discovered program state.

36. A method as defined in claim 35 wherein the second threshold and the first threshold are substantially identical.

37. A method as defined in claim 33 wherein there is no program state between the current program state and the previous program state.

38. A method as defined in claim 33 wherein the first, second and third thresholds are substantially identical.

39. A method of managing a cache comprising:

identifying program states associated with an executing program; and

if a first discovery time of a current program state is less than a first threshold and a first discovery time of a previous program state is greater than a second threshold, pre-fetching memory objects associated with program states having a first discovery time less than a third threshold.

40. A method as defined in claim 39 wherein identifying the program states associated with the executing program comprises:

developing a trace of the program; and

identifying the program states from the trace by comparing intersecting sets of data at least partially indicative of addresses associated with the trace.

41. A method as defined in claim 39 wherein the first threshold comprises a percentage of a first discovery time of a most recently discovered program state.

42. A method as defined in claim 41 wherein the second threshold and the first threshold are substantially identical.

43. A method as defined in claim 39 wherein there is no program state between the current program state and the previous program state.

44. A method as defined in claim 39 wherein the first, second and third thresholds are substantially identical.

45. An apparatus comprising:

an off-chip cache;

a trace sampler to develop a trace of a program;

a state identifier to identify program states from the trace; and

a short-lived object identifier to identify program states associated with short-lived objects based on the times of first discovery of the program states.

46. An apparatus as defined in claim 45 further comprising a cache flusher to remove a short-lived object from the off-chip cache when the executing program transitions from a program state associated with the short-lived object to a program state associated with a long-lived object.